many customers they signed that day, or how many packets they carried that day, or how many E-mail messages they carried that day, or something to just show that completely independent of and largely oblivious of all of this academic history stuff, this thriving, growing, crunching industry has sprung up by leaps and bounds. That most people in the Internet industry were not out of grade school yet when the Arpanet was being built and don't know a thing about it. Most people who are currently employed in the Internet service industry have graduated from college after 1990, which means that they have never experienced the Arpanet in any form and to them it's just a word.

Can you explain to me the evolution of FTP and Telnet and SMTP?

SMTP is an outlier.

An outlier?

It's 10 years later and much different. FTP and Telnet were for a long time the only applications on the Arpanet. They didn't so much evolve. They were both built in a weekend essentially.

Built in a weekend by whom?

Ken Harrenstien wrote Telnet. Although he probably will deny it, because the son of Telnet that he wrote, a program called SUP-DUP, is one that everybody who is anybody used, and Telnet was scorned by everybody except the original core of UCLA people, and all cool guys used SUP-DUP and MIT software with it. It was a Super Duper Telnet, but you only got six letters to name it in, and so SUP-DUP was it's name and Ken Harrenstien wrote it.

But it all evolved the same exact way that this terminal emulator thing that Bill Duvall wrote evolved. It's just, *"My God, here is this network. I can't use it because I don't have anything that will use it. What can I throw together as fast as possible so that I can get on the Net?"* And so Telnet was thrown together in a weekend. If Ken Harrenstien didn't do it, he will know exactly who, when, and where did do it, because it came out of MIT. FTP was probably written by John Goodtog(?), although I wouldn't swear to it. John Vital would know who wrote it. He was

certainly part of that crowd.

One thing I'm trying to get straight is that people use Telnet a lot and I guess FTP.

FTP, for the entire time that I used the Arpanet, I never used anything but FTP and Telnet. Those were the only programs that I used. Maybe Mail, but mostly FTP and Telnet. Originally there was a mail capability built into FTP and it wasn't until the late 70's that a way of doing mail outside of FTP came along. But us diehards just used the old way anyhow. It is impossible for me to over emphasis the primitive nature of FTP and Telnet. Let's suppose that you were living on a planet that was entirely desert and then somebody came along and invented oceans. *"Well, heck, I need a boat. What should I do? Well, here's a bunch of barrels and here's some rope. Let me lash these barrels together. Here's something that works as a paddle."* That's Telnet. Okay. How fast can I throw something together that floats and can be made to vaguely move where I want it to? It is just a very, very, very short-term, *"How can I use this right now?"*, 20 minutes of programming is 19 minutes too long, type of application.

It is impossible to over emphasize how little work went into the planning of Telnet. I'm quite certain that it was created in a weekend. But it has then had probably 100 or 200 modifications made to it after that. If you look at the RFC documents, something like 10% of all of them ever written have to do with Telnet. Because people kept enhancing it, adding this feature and that feature, and since it was the first tool, for many people it became to only tool, and as the only tool, it needed options and new this and new that. Whereas, other people invented other tools, some people just kept using Telnet. It's a software implementation of nothing. I mean, it gives you a piece of wire simulated with the Arpanet. I frequently call it a software simulation of a piece of wire. Yes, I can connect from here to here. I can use a wire or I can build an Arpanet and run Telnet over it.

FTP was somewhat more thought went into it, because of the fact that it had to move real data. So FTP had to cope with the *my word processor can't read your word processor* problems. FTP was the battleground in which the Sigma 7's of the world and PDP-10's of the world slugged it out. Because when all you're doing is sending key strokes back and forth with Telnet, everybody knows what it means to send a J. But nobody necessary agrees on what it means to send a word processor

document. Actually, word processors didn't exist at that time, but it's a good concept anyhow.

What would have existed that would be a good example?

Programs. If I wanted to send you a program that you could run or a mail file or image. In fact, the most difficult thing to send at that time would have been pictures.

Sure. How did that happen?

That was very hard and it required a lot of programming by careful people. But interestingly enough, the world's leading picture processing people at that time were at Princeton, and Princeton was not on the Arpanet. So, in fact, the Arpanet evolved without benefit of the best experts because they weren't on the Arpanet. You know, what do you do?

So you were saying FTP --

Telnet gives you a remote presence and FTP copies data. It's like telephone and Federal Express, if you will, and I'll call you, or telephone and fax. One of them is you can communicate and the other one is you can copy things that matter to you. Neither of them was designed. Neither of them was planned. They just kind of came out of, *"It is now Monday. By Wednesday, I must get this picture to MIT. I can't figure out how to do it. I'd better write a program. I think I'll call it a File Transfer Program, because what I need to do is to transfer a file to MIT."* And that's about the level of planning that went into it. FTP evolved in a much richer way than Telnet did, because of the need to copy complicated things like pictures and binary programs and experimental data and what-have-you. It's not enough to just get the letters there. You have to get all the fonts and whatever binary control information. Precision is more important when you're moving data, than when you're just moving letters. So FTP had a higher requirement for precision than Telnet did.

Berkeley Unix.

Berkeley Unix. Berkeley Unix, including TCP/IP software in it for free, was a major force in the growth of the Internet. Sun didn't contribute any more than any other

computer company to the commercialization of this, because absolutely all computer companies, even the bad ones, ran the same software. Since Joy was the guy who wrote the code at Berkeley and since he went to Sun, he is of course going to tell you that Sun is important, and I will not stop respecting you if you put that in your book. If you choose to believe that Bill Joy is telling you the truth, I won't leave you for it. But it's bull shit, and it's a piece of bull shit that he loves to tell people. He's said it so often that he probably believes it himself.

So the important part was Berkeley Unix?

Berkeley Unix was totally crucial to the Arpanet going from 1,000 nodes to 100,000 nodes, because the 99,000 additional nodes were all Berkeley Unix. But at that time, neither Sun, nor HP, nor IBM, nor Digital, nor Apollo, or any of the other companies that were shipping Berkeley Unix had anything better than anybody else. All companies just had their flavor of Berkeley Unix and that's what we shipped to people.

Brian just said the packet radio as a concept is very, very sound.

And the particular implementation of packet radio that I'm using here is something that Digital Cell has called a Roam About. You buy this little antenna and this little card to stick into your computer. It's $200. And you're on. It just works. I didn't even open the manual. I just plugged it in and turned it on and it ran.

[Tape 1, Side B ends.]
[Tape 2, Side A begins.]

Tenex is an operating system for the PDP-10 computer. That's where the *ten* in Tenex comes from. It's the ten executive. It was produced largely by the same people who produced the Arpanet. It was a BBN production. It came out of the same tight central control philosophy that if not only we could network all these places together, but in fact, they were all the same, then we could have great benefit to the world from all of the sameness. It was the Windows of its era, if you will. There was a 900 pound guerilla saying, *"This is what we make and therefore this is what you will use."*

And the guerilla was BBN?

BBN, right. BBN was the Microsoft of the 60's in this field, and they made Tenex, and they wanted people to use Tenex, and it was much harder to use anything on the Arpanet that wasn't Tenex. My primary association with the Arpanet was at three places, none of which would even think about using Tenex. I'm very strongly steeped in the *Tenex is evil* camp and this could have in fact carried over into my *Windows is evil* point of view in 1995. Not sure.

No. Windows is evil.

Yeah, Windows is evil. It's true. I have to use it here because I have no choice, but I don't like it. I'd prefer to pirate my copies of it whenever I can. Don't you dare quote me.

You want me to erase it?

No. It's all right. I trust you. So BBN wanted everybody to use Tenex for exactly the same reasons that Microsoft wants everybody to use Windows. There was a certain amount of work in Arpa to make Tenex the official operating system of the Arpanet. Carnegie Melon did not use Tenex. Stanford did not use Tenex. MIT did not use Tenex. Those were the three places that I did 95% of my Arpanet work. So all of my exposure to Tenex was as a visitor connecting to other strange places where they actually used it. What we found is that Tenex is what the non-universities used and universities -- in fact, I cannot right now think of a single university that used Tenex. There might have been one, but I can't think of that. Maybe USC or something. Tenex was used by the companies and other things were used by the universities.

I thought that PDP-10's all used Tenex.

No. There were three different operating systems for the PDP-10. Tenex was the one that BBN was pushing. But there was Tops 10 and there was Tops 20, and then Stanford and MIT both had their own operating systems for it that only they used. MIT's was called ITS, the Incompatible Timesharing System, and Stanford's was called SAIL, the Stanford AI Laboratories System. Actually it was called WAITS,

but everybody called it SAIL. There were four or five or six operating systems for the PDP-10. I have never worked at a place that actually used Tenex. My perception of it from my graduate student and then professor vantage point on the Arpanet was battle ship grey, military industrial, dreary thing that people used when they had to. My guess is that there were some advantages to using it, but having never worked at or visited a place that did use it, I've never been exposed to those advantages. But it was certainly official.

The Resource Sharing Executive was a UCLA thing that actually Carl Sunshine and his band created to -- if you hypothesize that everybody will be using Tenex, then you should be able to build a layer on top of that and somehow let people log onto the network without logging onto an individual computer on the network. To the best of my knowledge, nobody ever used the Resource Sharing Executive for anything real. It only was ever used in demos and playing around.

Oh, really?

If you have any evidence to the contrary, I would love to see it. But I never ever saw a real use of the RSE.

Do you think that Kleinrock's use of it --

Asking for a razor is the kind of demo that I'm talking about. I've never seen a situation where a person would sit down to begin their daily work and begin that work by connecting to the Resource Sharing Executive. It was just something that was there.

Is our description of it okay?

Uh-huh. It's small. Adding 10 words to it would be giving it more due than I think it's worth. This is an instance of a very intriguing phenomenon that has always perplexed me about the Arpanet. That is that the Arpanet itself was built by a band of world class people. There was just nobody better than them. But hanging around at their periphery was a whole bunch of second rate people whose chief qualification was that they had access to the Arpanet.

Who are you referring to?

And they kept inventing things that were bad, but somehow because they were connected to the Arpanet, the world was tempted to give them a certain amount of credit for this. There was a whole group at UCL -- I'm really reluctant to name names with the tape recorder on. [tape is turned off and back on] **The one who really comes to my mind as an example of this is Carl Sunshine. He and people that he worked with --**

Carl Sunshine, I don't mention him.

Okay. But he is an example. The whole RAM gang never came up with anything decent on the Arpanet despite years of trying. The whole software, tool works, or soft works, or whatever. There was some information programming institute that was dimly attached to UCLA that I'm sure was some kind of a faculty spin-off that got all kinds of publicity, and nothing that they programmed ever really was real. There was a company in Cambridge, Massachusetts called CCA, the Computer Corporation of America.

Tom Merrill's company.

Yeah. And they never did anything real as far as I could tell. I mean, they got massive amounts of publicity for what they did. But nothing they did ever penetrating any portion of the world that I lived in. You know, I would read these scientific papers about great strides that were being made on the Arpanet by CCA. But then the Arpanet, as I was a citizen of it at a time when there were only 500 to 1,000 of us in the world, I never saw any of this stuff. It's like they would announce it and talk about it, but then nobody would ever use it. The one exception was the Data Computer at CCA, which people actually did use.

What was the Data Computer?

It was an IBM TBM, terrabit(?) memory. Basically a video tape storage device. IBM had made it for large data centers like Social Security and CCA got enough money to buy one and connect it to the Arpanet for people to store data on. I had lots of my data stored on the TBM. But it's long gone. I think that's all I have to

say about Tenex unless you have questions. The people who worked on Tenex thought that it was very important and that it was at the very core of the Arpanet. But nobody who used the Arpanet and who wasn't paid to work on Tenex even liked it.

That comes up later in the chapter.

Mail and MLFL were two commands built into FTP and they both did the same thing, but they did it different ways. In both cases, they took a message from the sender and transferred it to the recipient, and they differed in where they got the message from. The mail command would take it from the keyboard and the MLFL command would take it from an existing file. So if you wanted to send mail to somebody and you hadn't already written it, you would go into FTP and type *Mail, space, person,* and then type it, and then it a dot in column one. But if you had already produced it with a Text Editor some place and you just wanted to send it, then you would go into FTP and type *MLFL, space, user name, space, file name*, I think. I've forgotten the exact syntax of MLFL. how shameful of me. But anyhow, they were both for sending. Neither one was for receiving. They differed only in where they would get the data from.

Arpa paid money to have mail programs created for Tenex. Arpa didn't pay money to have mail programs created for other things, because Arpa was pushing to have more uniformity than the community wanted, and they tried to control it by only doing explicit funding of mail for that system. What happened instead was all the universities looked at this and said, *"This sucks."* And they siphoned out of their umbrella Arpa contract to produce mail programs on the side and justified it as overhead and utility. So one of the big research universities produced their own mail system. None of them used Tenex. None of them could be paid to use Tenex. The wars about Tenex versus not Tenex were very much of a funded Arpa corporation versus unfunded university academic war. As I told you, many times I was on the university side of this. What I saw was you couldn't pay people to use Tenex. Those folks who worked at commercial research establishments that were being 100% funded by direct Arpa contracts used whatever they were told to use, because that's the way the system worked. But very few people who had a choice, which is to say the universities, ran Tenex. So this statement, *"Tenex was becoming the de facto standard and since there were more PDP-10's running Tenex than any other type of*

host on the network, most new E-mail programs were written in Tenex format." **That's probably true but very misleading, because I'm certain that Tenex did not have more than half of the Arpanet. It's probably true that there were more Tenex's than anything else.**

Then how should I put it.

Tenex was what Arpa was funding.

Which made it?

Which made it more likely that significant funded software development projects would take place on Tenex. But the life force of the Arpanet came from the universities and not from the commercial establishments.

Right. I'll put that in there.

Mail was layered. The fundamental principle that came out of the Arpanet was layering. An example of this is that Mail was layered on FTP, and FTP was in turn layered on NCP, and so --

What did we put there?

You talk about the headers being used to reassemble packets into mail messages, and that's not what they are for at all. When I have a mail message and I want to send it to you, I formulate it in its entirety exactly the way I want your system to get it. I hand it over to the mail transport system and then somebody somewhere chops it up into packets, sends them, reassembles them, deals with errors, and at some point, the mail message resurfaces on your computer as a file that is equal to the one that I sent. At that point, long after the packets are gone, the headers are looked at.

Are you talking about headers in the sense of what you, the reader, sees?

When you are talking about message group, mail headers, E-mail compatibility, Einar Stefferud, Dave Crocker, that kind of thing, the only kind of headers you are talking about are the things that in 1995 show up in the front of a mail message with

a colon in them.

Are you sure about that?

There are two issues here and they are very crisp. The issue of the first is how to interpret the characters and issue the second is what to do with the mail when it gets there, and there are no other issues. If I create a file on my computer, I can get that file to your computer containing exactly the same thing that it contained at my end, and the locating of your computer to choose a destination is done by me and not by you. So when I'm trying to send mail to Katie Hafner, one of the steps in that is that I have to know, my computer, my software has to know where on this big, bad network to send a mail file for Katie Hafner. But that choice is made by the sender and not by the recipient. So it is up to the sender's mail processing software to be able to decode network destinations from mail messages.

So if we assume that all that works and that I have made a mail file and my mail software has gotten that mail file to your computer, there are exactly two issues left. Issue of the first is what do the letters mean? If I have typed a capital W on my screen, does it show up as a capital W on your screen? What if my keyboard has a cent sign and yours doesn't? What if my keyboard has lower case letters and yours doesn't? How do we handle the fact that your keyboard and mine have different characters on them? We need to have some way by which we can agree on how to interpret the actual character codes. Having solved that problem by whatever technique, there is a second problem which is this mail file has now been transmitted to your computer. What exactly is your computer supposed to do with it? It is clearly intended to be delivered to some user at this site. How many users? Is it just one? Is it more than one? How do I process this message so that it can be delivered to that person?

The message group negotiations were over those two issues. One of them was headers of mail messages which are processed by the receiver to figure out how to handle the message when it has arrived, and then there is simply interpretation of character set.

When you say *header*, in what context are you using header?

Mail message header. The word *header* in the context of E-mail always means things at the front of the message like *To:* and *Subject:* and *From:*. The stuff that's in chancellor's message. His message is the classic example of a message that has all header and no content. But all this wrangling over message headers is about this stuff.

That's right.

Also, it is not just something that the user sees. Some of these are processed mechanically by receiving systems. There are about seven of these fields that have reserved meanings and very, very tightly defined syntax, and if you get it wrong, something won't work.

What won't work?

For example, the way you specify the date must be exactly so. If you put a comma in the wrong place, then your mail program's ability to check postmarks is wrong.

So it was the fight over the standardization of this?

Yes.

That was the fight.

That's correct.

And none of this.

The packet level stuff was three years old and had already long since been fought and nobody was going to open that again. Packet headers were ancient history by this point, and we all assumed in the E-mail world that the packets were going to get there and that some supernatural force was going to put the packets back together into a file. The basic assumption was that the recipient of this message would have a file that was equal to what the sender sent. All of this wrangling is over now what?

Right.

How does the recipient's operating system take this message apart, put it into the recipient's E-Mail box in such a way that the recipient's E-Mail processing program can understand it. The most common complaint had to do with the reply command. If you send me a message and I answer it, which one of those damn fields is in fact the one that I should send a reply to?

When I sent that message about the quasi-robot, I had totally forgotten about this until right now. This whole episode had completely left my mind. I now remember exactly how this worked. I had begun noticing that this Message Group was kind of like a social club. We had argued with each other so much and called each other so many names that we had become friends. I wanted to test the idea that although this idea was formed for a focused special purpose that there was a groupness to it or a sense of community that had evolved by accident along the way. So when this thing came up that I thought would actually be of interest to the majority of the people in the group, I sent it out as a conscious experiment in testing to see whether or not the Message Group had actually become a virtual community, and what I discovered was that it had. Yes!

The Message Group itself was not a standards making body. So of course, it couldn't come up with a standard. But the current mail standard that is in use today is RFC 922.

Is that SMTP?

It is the header standard that went along with SMTP. RFC 833 was the previous generation. RFC 922 supplanted that. In RFC 922, every single member of the committee that wrote RFC 922 was on Message Group, and virtually all of the contents of RFC 922 were submitted in the Message Group. Actually, that was Header People, not Message Group. I take it back. The Message Group, for some reason, most of us originals were on both Message Group and Header People. Something happened to move the raging debate out of Message Group and into Header People. I don't remember what it was. But in fact, it is true that Message Group petered out before the standard closed. But those of us engaged in the dialogue simply moved it over to Header People and kept going, and that was where

we reached convergence. My own mail archives only date back to 1980. I don't have anything that predates that. I wish I had saved them. But I have all mail since 1988 and most mail since 1980.

The difference between Message Group and Header People is actually another piece of important research, because Message Group was moderated and Header People was unmoderated. There was a respected elder of Message Group who all of us had agreed to listen to, Stef, and Header People was a free-for-all that basically was like an MIT dorm room brawl. Header People was the very first international flame fest. Although it was very annoying to be on and there were these long, unbelievable things that you had to wade through, that's where the action was. If you wanted to really influence the decision makers and be at the front lines, you had to be on Header People. The problem is that the tone of civility in Message Group ultimately prevented it from breaking new ground which was its intended purpose. So it just kind of went into retirement as being too nice to actually have the battles that were necessary to move to the next step. So the wars that produced today's mail standard were started in Message Group, but finished in Header People.

So the way to say it is that years later this all got resolved with SMTP in Header People.

SMTP is a protocol for sending mail and RFC 922 is a standard for the structure of that mail after it has been sent. SMTP is a formalization of the process by which you take a file containing mail and move it from one place to another. It separates it out from FTP in much the same way that we talked before about separating out TCP from IP. There are two pieces of the mail standard. One is the delivery standard and the other is the content standard. The content standard is all about headers. The delivery standard is all about how to get it there. SMTP is how to get it there. There is no name for --

So the thing that got standardized was the header style.

That's right. SMTP was created by one person. I believe it was Postel. It's really just a simple piece of network engineering and there was very little controversy in it.

And all he did was separate it from FTP.

That's right.

During this whole transition.

Yes, that's correct. And there might possibly been something controversial in SMTP, but I certainly don't remember any big controversy.

Okay. Well that's E-Mail.

What do I think you're leaving out? Not much.

What about news groups?

Not relevant. News groups were invented at Duke University in 1980. I actually own the original. I have in my file cabinet behind me here the original document by Tom Truscott(?), in which News Groups were proposed. News Groups and Arpanet didn't converge until about 1985, and by then, the concepts were separate. News Groups are important and it took the Internet to really universalize them. But this book has to have focus and the focus is about global computer networks and that's Arpanet. If there is one theme of this book, it is the virtual community, so far as I'm reading it. That's what I see you building up to here. The cute piece of recursion in the book is the describing the sense of communities in the people building the virtual community. That has a regularity to it that I really like. So one of the ways that you might be able to tie it together is to find some way to, in the early parts of the book, talk about the sense of community at BBN among the IMP guys. Then in the later part of the book, try to use as many of the same words as possible to talk about the sense of community of something that was built using the Arpanet by people who had never met any of the IMP guys.

Uh-huh. What would that be though?

Well, from what you've talked about so far, Message Group is your only choice unless you have SF Lovers archives. The four big lists that I can remember are Message Group, SF Lovers, the telephone group whose name I can't remember. It's currently run by John Solomon.

At USC?

No. I don't know where he is now. All of these thing originated at MIT. It was the center of the social universe, which is weird considering its reputation. Oh, Telecom was the name of it or Telecom Digest. Then there was another one that was unbelievable in its time and it was officially shut down by its owner the day that the Soviet Union fell. It was called Arms D. There was a chap at MIT who then went on to be on the staff of Senator Proxmire and who now works for the National Academy of Sciences names Herbert Lin, L-i-n, who ran the first social science discussion group on the subject of arms control. The arms control mailing list, the science fiction mailing list, the Telecom mailing list, and the Science Fiction Lovers were the four big ones that I remember.

When did Science Fiction first start? Do you know?

It was already in existence when I started paying attention to Arpanet mail in 1975.

That's interesting.

[End of tape.]

BRIAN REID INTERVIEW

Used ultimately to support human to human communication for which no mathematical modeling is possible. So what Kleinrock did is he claims to have analyzed computer networks. But what he really did was he formulated models for computer networks and then analyzed them. So the analysis was not of the network itself, but of a supposed mathematical model for the behavior of the network. For many years, his mathematical models went completely unchallenged because he was so good at math that there were very few people who knew enough about computer networks and mathematics who could challenge his models. So his papers would contain such things as, *"Let us suppose that the packet arrival behavior of a computer network can be satisfied by the following equation: [makes noise]."* Page full of equations, you know. You'd just sort of look at it and think, *"Heck, I can't understand that."*

Must be right.

Must be right. And then he would analyze the dickens out of it and publish a paper and some student would get a thesis out of that analysis. And it wasn't until 1985 or so, when two Stanford students, Jeff Mogel and David Boggs, who are both very pragmatic, very nuts and bolts oriented. They know theory, but they tend not to be influenced by it. Besides, they would actually measure some of Kleinrock's assumptions just to see how well his model fit the reality of computer networks. And what they discovered was that he was measuring the wrong thing.

Because the reality of the network is such that --

Is much more random and much less predictable than any of Kleinrock's models. But the random reality is unanalyzable because it takes human behavior into account. When people's fingers are moving on the keys, how often do they send things? Or when nine people are all sending E-Mail on the same topic, and they are all hungry, do they all focus on finishing by lunch? It really isn't random. Kleinrock's fundamental assumption was that he could come up with a simple mathematical expression for what people wanted to do with computer networks and then a complicated analysis of that expression, and on the basis of this, he has been for years publishing assertions that Ethernet had the following limit, *"Ethernet was only good for thirty percent of capacity. Computer networks would saturate in following*

the certain circumstances." And had been proposing alternative after alternative for how to go beyond them.

Luckily, Ethernets were cheap enough that people decided to ignore him and go ahead and use them anyhow because they were only thirty dollars or something, and discovered that it wasn't true and that all these predictions of saturation simply didn't come true. But he's so smart, so glim and so pervasive that there's a whole generation of engineers out there, who if you ask them, will tell you, *"Oh, yes, Ethernet has a thirty percent load limit. And, if you try to load and Ethernet with more than thirty percent of it's capacity, then it will collapse.* And that's complete nonsense. But, if you believe the theoretical model of an Ethernet, and the way human beings will use it, then that's not nonsense. And it was many, many years before people got the ability, the time, and the motivation to challenge his models. There was a very famous paper presented by Jeff Mogel at a 1987 conference in networking where Mogel just went through each one of Kleinrock's positions and showed that it was bologna.

Kleinrock decided to focus on Mogel's math instead of on his results and my guess is that to this day that Kleinrock probably believes that Mogel is wrong and that he has made this or that mistake. But the world has voted with its feet, and people now use Ethernet as the basic building block for everything in spite of all of Kleinrock's published predictions that it won't work.

Start with Steve Crocker's.

The way that Steve Crocker wrote RFC-1 was extremely important in setting up the social and emotional reasons for the success of the Internet, because it was written in a style that was not imperious, not heavy handed, but cooperative. When you read RFC-1, you walked away from it with a sense of, *"Oh, this is a club that I can play in too. It has rules but it welcomes other members as long the members are aware of those rules."* I remember when I read RFC-1, my overwhelming reaction was, *"Oh, this is good. This is inclusive. They want my help."* Whereas, similar network documents from other protocols have been, *"We're the protocol Gods and you're not." We say how it's done and then you do it.*

Like these 802 committees?

Yeah. Like the 802.3 committees, for example. And the thing about RFC's and Steve Crocker's creation of the concept was that it brought along a style of cooperation and ego reduction. Because Crocker did not put his ego in that RFC and it was number one, it set the style for the next thousand RFC's to be egoless, cooperative and friendly. It is impossible to underestimate the importance of that, because what it did is it made newcomers like me, who didn't come along until three years later, feel welcome to join this club. I did not feel excluded by a little core of protocol kings. I felt included by a friendly group of people who recognized that the purpose of networking was to bring everybody in. And when I came along three years after RFC-1 was written, the social tone that it set was strong enough that I felt welcome and people like me felt welcome. I believe that is the biggest single reason why the Arpanet succeeded so powerfully is that new, smart people who were not part of the original core were made to feel welcome and that their contributions mattered.

Oh, that's great. That's great.

There's two different ways of looking at the hardware world. There's the system view and the component view. Wingfield was a component guy. What he did was absolutely crucial. More crucial than I get a sense so far from what you've written. The IMP was a box that was built from whole cloth. It was a hardware design to produce a hardware device that you could look at and touch and pay for and lift off the ground. It was an object. But what Wingfield did was to make a little piece of electronics that nobody could see that would go inside the computer, whose purpose was to correctly talk to the IMP.

Inside the host?

Inside the host computer. And as such it didn't really have an identity, nobody ever saw it, it didn't have an asset tag number, so when the inventory people came around, it didn't show up on the books. It was essentially invisible. If you look over there on my bookcase, you will see a bunch of interface cards. In fact, if you want to walk over and pick one up and fondle it, you will understand what I'm talking about. Pick up the top one and bring it over here. That one right there. That is a modern interface card, and that particular one that you're holding there is a T3 interface card for a digital computer. It was designed by David Boggs and it's

brilliant. There's nobody else alive who could have made it. The fact that this fits on one card causes this computer to have much greater use than it could possibly otherwise have. But when I put this interface card into a digital computer, the card becomes invisible. All you see is that little connector right there sticking out the back. The card is subsumed into the computer and becomes philosophically part of it. But without this card, the computer simply cannot do T3 communication. But the minute you install this card, the card's identity disappears. So the design of interface, what this card does is it takes concept A, namely T3 communications and puts it into concept B, namely fast work station --

So, in the case of the IMP to host interface, host to IMP interface, it took concept AB --

IMP and put it inside concept B which was sigma seven. The Arpanet was utterly dependent on those interface cards. I can remember being at Stanford in 1980 and being desperate to get my hands on a host interface card and that there was some little company down in Southern California that was about to go out of business that was the only people alive that could make them.

ACC?

ACC, that was it. And what can I do to keep ACC afloat long enough to make that card, because I need it. *"Would you like one of my children, would you like my car? I don't care, I have to have that card."* But the people who design these cards in general get no respect, because their function is to be invisible. The very best hardware designers that I know strut their stuff to each other by building interface cards. It's the ultimate macho in that kind of hardware design. But nobody outside of their own little group understands what they do.

Well, you can actually extrapolate that and take it one step further to the Subnet, to the Subnet of IMP's.

Yes. That's a good point. That's a good analogy and it proves to me that you understand it. Because in fact, the purpose of the IMP Subnet was to be invisible. That was not a hardware design project, that was a systems design project. But the philosophical problem with it was the same. Is that when it is totally successful it is totally invisible and you forget that it is there. Only when it fails are you aware of

it. And so if the people have done a good job, then you forget all about their work. It is an occupational hazard of people who build mechanisms to join one world to another. Because when they have totally succeeded, they become invisible. So you cannot succeed in that business if you have an ego. If you need to be famous like Kleinrock in order to be happy, you will never make it. So what Wingfield did is every bit as important as what Ornstein did because he produced a device that joined an alien world to -- it brought the concept of IMP into the concept of sigma seven seamlessly. And the sigma seven looked at it and said, *"Oh, yes, I recognize this. This is an IO card."* And the IMP looked at it and said, *"Oh, I recognize this. This is a 1822 card."* And each one of these two machines was able to see this thing as native. And, it successfully balanced between them without being able to change either one. It had to adapt totally to the requirements of the sigma seven. It had to adapt totally to the specification of the 1822 and couldn't ask for changes to either. They had to be taken as constants. That's very hard to do right and it takes a real genius to do it.

Good. That's important.

What you're writing here is the story of the growth of a network. There's one point of view that says that the network has a single center and that the early days of the network are the early days of that center. But in fact, the thing that is revolutionary about the Internet is that it doesn't have a center and it never did. It's a peer to peer communication.

You're saying Internet?

Arpanet, Internet, same difference really. The Internet is just the successor to the throne of the Arpanet. The Arpanet was peer to peer communication, and every member of the Arpanet had a technical and social status that was essentially the same as every other member of the Arpanet. And so when the Arpanet grew to include the 10th place, all of the people at that 10th place had a story too, and until the size got to be about 50, each new addition gave you something wonderful that you didn't have before. So all those people really were at the beginning. But the thing that they were at the beginning of was the next phase.

What do you mean that each addition gave you something wonderful you didn't have

before?

You had access to every -- the first 50 places that the Arpanet connected were all completely unique. There was no repetition. I don't know. 50 is probably too big a number. The first 30 places or the first 20 places. When it finally went to Carnegie Melon in 1972, Carnegie Melon had an artificial intelligence department bar none. And when it went to Stanford at about the same time, Stanford had the robotics group. So what you get when you put a robotics group on the Arpanet is something very different than what you get when you put a communications performance center on the Arpanet. Because what you are doing is you're bringing peer to peer communication to a group that hasn't had it before. And the wonder of the Arpanet is not what it is, but what you can do with that. The reason for the success of the Arpanet has to do with the fact that it enables people to do their job better.

Therefore, the real story in the real Arpanet is not about the mechanism, but about what you can do with the mechanism. And so the very first time that deaf people were put in contact with each other in the Arpanet, there's a story. The very first time that a robot in one state was controlled by a robot meister in another state, there's a story. And each of those stories is a first. And the people in it feel as though they were right there at the beginning. Because the thing that they were at the beginning of was this horizontal wave of networking and this vertical wave of technology crossing at their place. And that's the reason that you've got so many names in this is that this all happened so fast, so geographically dispersed that there were hundreds of firsts all taking place in the same half decade, and the people are absolutely correct that they were there when. And yet, you can't write a book with 5,000 characters in it.

So, what do I do?

That's the way technological growth happens. I defy you to name a technological innovation whose first use was not copying something old.

But the point you were making is that the Arpanet was to -- it just seemed a little strange to me that the Arpanet was being used to --

To implement client master slave communications with a teletype. But that's what people understood. The way you trust new technology is to demonstrate to yourself that you can use it to do something that you understand. That's how you get comfortable and safe with it. And then once you've done that, then you can take the next step and say not only can we emulate a terminal, but we can do things that we could never do before.

Well I guess what I'm asking you is should I take that out?

No. That is a very key concept in the entire book.

I came up with it all by myself. I'm so proud. It just dawned on me. And it is a key concept.

It is a major key concept.

So maybe I should expand on it a little.

If it were me, I would expand on it at the expense of some other details.

And what would you say?

The mechanism of how it was built.

How what was built?

Oh, you know, how the login program was built.

I can't. They don't remember. I've been all over the map looking for these guys. Actually that is not how I would --. I would expand on it in talking about the implication, not the implications, but the bigger picture stuff that you just said that that's what technology does.

Yeah. With technology growth, when the horseless carriage came along, the initial use of it had a glove compartment, because when you rode a horse you needed gloves.

Oh, that's where that came from?

Yeah. When you had a carriage with horses, you had to use gloves for holding the reins or your hands would get sore and carriages had glove compartments. And so when you made a horseless carriage, well because it was like a regular carriage, it had gloves too. I am really and truly unable to think of any technological innovation that didn't begin its life by being used to imitate something older.

And then gradually what happened? And then after that --

People assimilate it and start to think about it and then they come up with the next generation of ideas.

And in this context, in the context of the Arpanet, people gradually started using it for --

Distributive computing and remote access to files. A very minor tweak on the concept of master/slave is client server, and so in fact, a terminal emulator is not necessary a master/slave relationship. Because a master/slave relationship requires that each machine be either a master or a slave, but never both. And the whole client server paradigm of computing, which is arguably one of the big things to come out of the Arpanet, is anybody can be a client and anybody can be a server, and if I need something done and you know how to do it, then at the moment we have a client server relationship where I am the client and you are the server. But then if you need something done and I know how to do it, then we turn it around.

And so the first remote terminal session between SRI and UCLA was you couldn't tell by looking at it whether it was master/slave or client server, and the real key test would come from turning it around and doing one over the same link in the other direction.

Which they didn't have the hack for. Did I put that in there?

No. You didn't tell me that.

Which I found ironic.

It is ironic, but it's the path that technology growth takes. And so the real breakthrough in terms of intellectual contribution came not with the first such session, but with the second in the other direction. And I don't know how much later that one came.

I don't either. That's a good question. I'll find out.

And we take that symmetry for granted these days. When SRI is the slave and UCLA is the master, then you are testing the mechanism. But then when you turn it around, you are testing the concept.

The whole deal of assembly language programming has to do with whose turf you do it on. These days programs are written in languages that are designed for programmer comfort, and the main purpose of a program, if I'm holding it in my hand, I'd say to you, *"I have here the source for Microsoft Word."* Which I don't. But if I did, I could say that. The language in which it is written is a language designed to be comprehensible to people and not to computers. Because it's primary purpose is to document the algorithm, and there are very complicated, very sophisticated programs called compilers which translate programs for people into programs for computers. And there are all manner of benefits from using compilers. But, in those days, compilers weren't good enough, and you couldn't really get your hands on the gory details of the machine. So when you wrote a program in assembly language, you were doing it entirely on the computer's terms. You communicated with the computer in its language. End of story. No human language involved.

It's language being?

Whatever it is. A computer is an instruction following mechanism. All computers are. That's what they do. They follow one instruction and then they follow another one and then they follow another one and so on. And if one of the instructions is to return back to step five, then you have a loop. But the essence of what a computer is, it is a device for following instructions. And when you write a program with a compiler, you specify the behavior that you want in a reasonably abstract way, and the compiler produces computer instructions from it based on the compiler's knowledge of the computer. When you program an assembly language you are dealing directly with the raw instructions by the computer and you put numbers into

it that stand for things that you know the machine knows how to do. And so you are really dealing with the computer on its terms and not yours.

And how did that make Crowther's job harder or easier?

Well, it's meant that Crowther -- . It's like traveling to another country and thinking in that language and not your own. In order to be able to program successfully in assembly language, you have to completely reshape your entire thinking process so that the way your mind perceives sequential action is in terms of the steps obeyed by the machine and not in terms of the steps that you are trying to accomplish.

It's as if, let's supposing that I were giving you instructions on how to go to the ladies room. To specify it in a higher order language, I would say, *"Go out the door, turn right, walk down just past the elevators, and then go left."* To say the same thing in assembly language, I would say, *"Put your left foot in front of your right foot, and then put your right foot in front of your left foot, and keep doing this until your left foot gets to within 12 inches of a cardboard box. At that point, stop and make a turn 90 degrees to the right, and then put your left foot in front of your right foot, and put your right foot in front of your left foot and keep doing that. But every time you do it, stop and look to the left to see if there's something that might be an elevator. If there's something that might be an elevator, then stop to check to see if it is an elevator, and then if it is an elevator, then resume putting your left foot in front of your right foot,"* etc.

Assembly language programming is like that. It is what seems like maniacal dwelling on the mechanism, so much so that you often lose track of the goal. I probably gave you three hundred words for instructions for how to get to the ladies room and it might be that you are so caught up in, *"Well, do I need my left foot or my right foot here?"* that you would forget that in fact you were headed to the ladies room.

But, Crowther didn't get --

Crowther is a tricky guy and he knows how to warp his mind back and forth between forest and trees. The amateur programmer in assembly language will completely lose sight of the overall goal while focusing on the mechanism. And when

you look a piece of assembly language code that was written by someone who is less worthy than Crowther, what you will find is aimless wandering. You see a little sequence of instructions that do this, and then you see a little sequence of instructions that do that, and you think, *"Well, how do we get from here to here? What's the relationship between those two?"*

The way Crowther works is he would just -- his psyche is big enough that he can assimilate the entire thing into his head at once and think of the assembly language program in its entirety, rather than focusing on the individual steps of it. And so in a certain sense, he is a human compiler. It's a very rare skill. I've known two or three people in my life who can do that.

And he's one of them?

And he's one of them.

And how is it that the fact that these guys programmed in assembly language relevant to this story?

You cannot make an assembly language program -- well, it isn't, except that assembly programmers are the ones who find bugs in hardware. And so in terms of your story, all relevant programs at the time were written in assembly language because compilers weren't good enough yet. I would be willing to bet that the Honeywell 316 had never been programmed into anything but assembly language. Because military computers were always programmed in the way to get the last drop of efficiency out of them.

Then there was the constraint they had, the memory constraint they had, which what was it? 16 --

16K.

It says 16K words. Because there weren't bytes back then. Right?

That's right.

That was a huge constraint, wasn't it?

It certainly was.

They had to write very, very --

Small snippets of code.

And that took a certain talent?

It took an amazing talent. It's somewhat like the -- let's suppose that what you were trying to do was design a piece of furniture that would be assembled by the customer in their home. *"Buy this sofa and then we'll ship it to you as 19 parts, and then you can put it together with just a screwdriver."* Imagine if you had never -- if it were not possible for -- you were doing this in a space station, where you simply didn't have room to test your assembly of the sofa. You could analyze each part individually, but you had no way of actually testing your belief that these pieces could be put together and made a sofa, and that the only way to find out if these pieces really did make a sofa was to sell it to somebody and then see if they can put it together. When you are trying to make an overall system work in terms of little code snippets, no one of which is bigger than such and so, in a certain sense, you are trying to design a large component out of small pieces, but it's not possible for you to test to see if they work together until you turn it on.

Well, which brings me to the other whole -- . There's a big problem with this jack.

I'm giving you a 90% confidence level on me. What I'm speaking of here, this is one of the first things that I will have ever told you where I am speaking from general principles of the industry and not specific knowledge of the situation. I was not there and I don't know this for certain. What I'm telling you here is informed analysis. The difference between the ruggedized final IMP version of the 516 and the prototype version of the 516 was twofold. First was the actual ruggedization itself and the second was the special hardware for the host interface. The special hardware for the host interface can only be tested by testing it. And that wasn't in the programmers development machine, I believe. But everything else was. If you could have a simulator or a stub or some stand in for the host IMP interface, the

IMP side of the host IMP interface. I don't know what it was called. But there is a piece of hardware in the 516 that was what connected to the wire that went to the host that was the IMP end of 1822 interface. That was special hardware made by Honeywell and there's no way you can test that until you have that hardware. But all of the other logic, all of the packet processing, all of the phone line drivers, the modem handling, everything having to do with IMP to IMP communication on the implementation of the subject itself, could be tested on the development machine.

So the thing that they couldn't test on the development machine was the host communication and the timing. They could make reasonable approximations to the timing, but in those days every machine was different in terms of how rapidly things happened and so you couldn't make absolute assumptions. You could merely have high probabilities. You know if you thought that it would take seven microseconds for this sequence of instructions to execute and it actually took 7.2 microseconds because of this and that timing situation, there's no way to predict that short of measuring on the actual hardware. In these days of standardization and everybody using the same brand of CPU's, the Intel chips, if I tell you that I have a 80 megahertz Intel chip, that gives you a very solid understanding of how many instructions per second it's going to be able to execute. But in those days, there was no standardization of the CPU's were all hand fabricated and they ran however fast they ran. No two were quite the same.

Well, I'll work on this section. So most people were programming in assembly language, but being a really good assembly language programmer --

The thing that was different about this is that it had really tight time constraints, and was also inner driven and was symmetric and not master/slave. So they actually solved the interrupt problem with something that I've never fully understood called the pseudo interrupt device or PID. I've studied the PID until I'm blue in the face, but I honestly don't get it. But it was a way of getting around the interrupt driven problem to make the stuff actually work at all by cutting the code up into snippets that ran so fast that they didn't need to be interrupted. And it is the one piece of the IMP structure that I can spit back to you, but really honestly don't understand.

How about the fake host concept. Are you familiar with that?

Tell me. I might be, but not under that name.

TCP stands for Transmission Control Protocol. Control is the operative term here. TCP assumes that you have a way of getting data from one place to another and it is a way of bringing order to that. It is a method for reconstructing sequence and reliability on top of a network that is inherently unsequenced and inherently unreliable. Packet switching systems sometimes lose data and packet switching systems sometimes send data out of order. Many people have a need not to lose data and have a need to get data in order, and TCP is nothing else. TCP is wonderful. Don't let me get it wrong here. But there's nothing to TCP besides a set of behaviors for the two ends by which they will not lose or mis-sequence data.

It's the reliability part of the --

Reliability in sequencing. It adds reliable sequence delivery to a transport mechanism. But TCP is completely unable to care about inter networking or how the data get from one place to another. You can run TCP over two tin cans on a string. What TCP is all about is a set of behaviors that the two ends have to exert in order to move data correctly, and TCP assumes the existence of a transport mechanism underneath it that moves data most of the time that is mostly right.

Which is IP.

Which is IP or old Arpanet or anything else.

But there was while when TCP and IP were all together and then IP got separated out of TCP.

No. Never.

Yes.

Factoring out TCP as a protocol in its own right separate from IP certainly took place during the design process of TCP and IP. But that was part of the John Postel, Dave Clark development of the next generation protocols. And in the beginning, it's true, nobody thought to separate out TCP from IP from anything else. But long

before anything called TCP was announced outside of the ten people who were working on it, they had this one solved. So this is an issue kind of like Frank Hart saying no.

However you know, this is the thing --.

In fact, I would be willing to bet you that there was not one single new idea anywhere in the IMP, and one single new idea anywhere in everything up through and including electronic mail. What it was, was a responsible, reliable, clever use of existing ideas in a way that made sense in the whole through the systems integration effort. It is absolutely true that lots of ideas were bread boarded in lots of places before the Arpanet needed them and that many of the places that were doing that bread boarding of ideas were on the Arpanet. There were just as many bad ideas as good ones. In fact, there were about ten times more bad ideas than good ones. And so while it is true that John Postel did not invent the separation of TCP from IP, it is true that he recognized the necessity of it when he saw it. I give him more credit for standing behind the idea than I do John Shock for having it in the first place. Because although Shock did have that idea at Xerox, he didn't know whether it was a good idea or not. He didn't take a stand. He put it out there to see what people would do and so in a certain sense he invented it by accident.

And the thing about Postel is, in his own very quiet way, he's really, really firm in what he believes in, and he really takes a stand.

Oh yeah. Postel is a rock. He stands for things. And that was his contribution. As he would sift through all this crap, then he would say, *"This is what we are going to do."* And God save you if you wanted to change his mind. And that contribution of sifting through one hundred candidate ideas and coming up with the one that was right was what Postel did. That was his role. And it just so happened that the candidate idea that most closely resembles the final product is John Shock's pup. But Shock himself didn't understand it when he came up with it. It was just another prototype.

Right. That's very interesting.

Now David Boggs, who was a graduate student at the time, understood it much

better than Shock did. And Boggs is probably the conduit by which ideas about pup reached Postel. But Bogg's role is he was a hardware guy. He wanted to build more pup hardware because he thought it was cool. He wanted to increase the number of people in the world who would be using pup hardware and so he spread the ideas all over the place.

Do you know the differences between NCP and TCP well enough to tell me what you think the major differences are?

Yeah. They're major differences are almost entirely having to do with, well, NCP was the integrated protocol that didn't have the layering and NCP stood for a network control protocol, and it was simply the name for all the aggregate things that happened inside the IMP sub network from host to host, and NCP was everything between one host and another taken care of for you by the IMP software. TCP and IP were conceptual layers that gave the same service through partitioning, in order that you could factor out the IP from the TCP. Yeah, TCP had some new ideas in it.

If you sit down and look at the specification of TCP versus the specification of NSP, it's a lot like the difference between Windows 95 and Windows 3.0. They are very clear evolutionary relatives. One of them, TCP is two generations downstream and it has more ideas and more features and more options and more capabilities, and while they were at it, they separated out the packet transport in routing.

What was the reason for doing that?

I've heard two stories. The decision was made by Clark and Postel. They announced their decision at a operating systems conference at a seminar in either '80 or '81. I forget which one. It was the only conference I've ever been to that had all the players at it. There was a whole session devoted to this very topic. Since everyone's ego was on the line, there were lots of conflicting points of view. I believe that it was done for the purpose of decentralizing management. I believe that Clark realized that the Arpanet required the BBN Network Operation Center in order to survive, and that nothing significantly larger than the Arpanet could be run from a single point, and that they needed some structure that would allow the Arpanet to be built out of smaller, more manageable, semi-autonomous units with a well defined

protocol for them to talk to each other.

I'm not sure of exactly how they reached this conclusion, but it was pretty obvious to everybody. I mean, I was in Pittsburgh at Carnegie Melon at the time, and it was obvious to us that you just plot the growth curve of the Arpanet. It's going like this, and sooner or later, it will be too big to manage. So what are you going to do about it as it gets to be too big is a serious research problem. So the real reason for making any change at all had to do with growth management and the strategy seemed to us to be -- Clark made the strategy and I've asked him this question and he was not able to articulate the answer. So it might be that you will never find the true righteous answer to this question. If you've asked both Postel and Clark, and they can't tell you, then it's lost.

Ask them why?

Our perception in the networking group at Carnegie Melon at that time -- we were on the mailing lists. We got all the notes and everything. Our perception was that the dominant force in their design process was growth management and the dominant strategy was divide and conquer. That the basic separating out was to make it so you could have a TCP implementation from one group using an IP implementation from another group over a network managed by yet a third group.

When you say growth management, do you mean the fact that Arpanet could only grow up to 255 hosts? Is that what you are talking about?

Oh, no. 257 nodes is still manageable by BBN. But 257,000 is not. So it's real easy to take a protocol like NCP and extend it to more than 255 hosts. The question that you need to ask yourself is while we're in here fixing things, what is the next thing that's going to break after the 255 limit? And can we fix that while we're in here too? Can we do some prophylactic fixing if you will? Can we fix things before they break, instead of after? And so as they looked at the growth curves and realized that in fact breaking through 255 was the least of their worries, but breaking through a million would be the next worry. What is it that you are going to have to do to the fundamental structure to cope with the fact that this number is going to spin out of control over the next five years? And I've never seen anything written down about this, but the conversations, the E-Mail, the table talk at the time was

that anything bigger than about a thousand nodes could no longer reasonably be managed by BB&N.

So a lot of it was the BBN kind of control freak issue, which they were.

Yes.

Getting back to that.

That's right, and everyone, except possibly BBN, realized that the central control freak concept didn't generalize. You can't go beyond a certain number of nodes. There exist some number of nodes such that if you have that many, then the BBN model fails completely. You put the Pentagon in charge of it. You shoot people who break the rules. I mean at some point you can make central authority work to a point, but sooner or later you have to realize like the Soviet Union did that central control is a bad idea for some situations and you need to bring -- is it Perestroika? Is that the Russian word for the decentralization? No. Perestroika was the name for regionalization of power and that was really what happened in the Arpanet is that the central control by BBN model had to go.

This was recognized when, and by whom, and toward what?

I started hearing it in '78-'79 time frame. The first person that I heard it from was Bob Kahn. It was kind common conversational topic in the Bob Kahn, Vint Cerf, Duane Adams, John Postel, Dave Clark crowd in '78-'79; that arena, time frame.

The topic being that BBN was going to have to --

Not just BBN. That the centrally managed, tightly controlled network model was not going to last. None of them said anything bad about BBN. But all of them said that central management by anybody wouldn't work and it was obvious that they were referring to BBN, but no one was slamming BBN particularly. They were slamming the whole concept of central control over a large global network.

But what were people thinking should happen? That it should go to whom?

Well, that was the research question. That's why this was being asked in the universities. And the answer that as far as I can tell Clark and Postel are the authors of is that you invent the concept of the autonomous system, and within each autonomous system, you allow them to run it however they want, and you write down protocols for the exchange of data between autonomous systems.

When you say autonomous, do you mean a system that doesn't necessarily have to be reliable?

It means not under your control. Your description of the host computers, each one of them believing that it was the center of the universe, is very apt. Because that is just what it was like for networks. I have in this house a network and I own it and I control it. It is my autonomous system and I get to decide everything about it. My next door neighbor might also have such a network. If we join them together, then at some point in the wire between here and my neighbor's house, there is a control transfer. It's stops becoming my turf and starts becoming his turf, and unless he and I are willing to negotiate between us all the time, then what we need is to have some standard set of agreements that we both follow for how to talk to each other. In much the same way that when I go to your house, I knock on your door and I wait for you to answer. That's the autonomous household protocol. I am my house. You are your house. When you go to your own house, you just walk in. When you go to my house, you knock and you wait for an answer. Then if I don't answer, you try knocking again. If I don't answer, you try knocking again. And depending on how desperate you are, you might try knocking on the back door or whatever. But sooner or later you give up and go away and you don't break in. Because the protocol says that if you can't get in, there is probably a reason. But in any event, you shouldn't do it because it is not your house.

So the whole idea that the network would be divided into independently administered units that had their own rules that were run by their own people, and the standard is silent on how you run your network. It doesn't care. But the standard instead talks about what you do when it is time for your network to exchange data with somebody else's network. And the purpose of that was to allow the network to grow by having a larger number of computers in it.

[Tape 1, Side A ends.]

[Tape 1, Side B begins.]

They were not comfortable that they knew how to manage something with 50,000 computers in it. So one of the vague targets was a few hundred to a small number of thousand computers is a nice number for a network.

So when you say units do you mean networks?

I mean administrative entities.

And were there such networks already?

At the time this was happening, there were exactly two such networks, and BBN was trying to build a third. It was Milnet and Arpanet. Arpanet had separated out from Milnet at -- I forget exactly when but --

'83.

Oh, no. This took place earlier than that. This was all in the '78-'79 time frame.

Which was the official switch over from NCP to TCP.

Yeah. The same time, approximately. It was like February 1, 1983. Something like that. They passed out buttons.

I survived the --

Yeah, those buttons. So it might be that this topic was originally raised as part of the Pentagon planning process for separating out Milnet. What I was doing, I was a user of the net. I was not a planner of the net. But I was a very demanding, concerned user of the net, and I made it my business to know what these decisions were.

-- DCA.

Yeah, that's right. I remember that.

But BBN was still --

It was scary.

It was scary, why?

The DCA guys were all spooks. In '75, there was this blizzard of memos from generals about things that you were and weren't allowed to do with the network. All of these rules and none of had any idea whether we should ignore them, laugh at them, or obey them in fear of our lives, because none of us had ever gotten any message from a General before and all of a sudden at least once a week, there would be E-Mail from General such and so, in all uppercase, and all this military speak about rules for the correct use of the defense communication agency resources, and us university guys had no idea what to do with information.

I can imagine. Alex McKenzie was telling me these funny stories about having to come up with forms. And one crazy conversation he and Dave Walden had with one of these guys from DCA when they sent something, but not by the proper channel, maybe by E-Mail or something. It was a proposal and the guys just through it away. Not because of what it said, but because it wasn't --

Because of how it arrived.

Right. That doesn't surprise you?

No. That's exactly. This was my first taste of big time government red tape landing right in my own front yard, and we basically made a big joke out of it with one year at Halloween when I would say 15% of the graduate students dressed up as Colonels.

Seriously?

Seriously.

Because of this?

Because of this, yeah.

So anyway, where were we? Oh we were talking about administering of what networks there were. Were you aware of packet radio and packet --

Oh sure. Jeff Goodfellow and all of those people.

What do you remember about it?

What I remember is it was a cool idea, very expensive, and it went nowhere.

Which one do you mean? Packet radio?

Yeah. I know them both. The Norjer(?) net. The Packet satellite stuff was largely in Norway and the packet radio stuff was at SRI. In both cases, it was intriguing and it got some published papers and nothing ever happened with it. We looked to see why nothing ever happened with it and came to the conclusion, that was actually, my analysis of why packet radio failed, is a main component of my career choice. Because I was in graduate school at the time and I was working on a thesis topic and when I saw that packet radio had met all of its goals and still nobody used it, I went to look at why and I discovered that it was not universal. You couldn't use packet radio to communicate with anything that wasn't packet radio. So it failed the compatibility test. They did not succeed in integrating packet radio in any usable way into the larger structure.

Larger structure of?

The Arpanet.

So is it packet radio that just got them thinking about Internetting?

I think so. The people who built packet radio were small picture people. My main contact was Goodfellow. I don't know if you've talked to him at all.

About other stuff.

And as you know, he's a flake, a visionary, and doesn't need the money, and he never took it very seriously. So finding out from him what his goals were, what his

groups goals were, was very hard to do. It seems to me that his goal was to spend government money buying cool radio toys.

And they have this SRI van going up and down the freeway. Do you remember that?

Yeah. Oh yeah. Jeff had one in his car, too.

One of these radio units?

Yeah. But the SRI van on the freeway was a fixture. Jeff loved to make sure that every body knew about it. He was always telling stories about the places that he went with it.

What did it look like? Do you remember?

It was a white panel truck.

I heard it was an old bread truck. That it had a lot of racks in it which was perfect for components.

I never went inside.

Oh, I think Shock told me that. Then, packet satellite?

I have communicated with people who used packet satellite. I've never met anybody who worked on it. But what I learned from packet satellite was that latency matters,

You mean delay?

Delay. And that any packet switching system that has a half second delay is going to be a failure. And I did not ever once in my whole career see any use of --

[Phone rings.]

Packet satellite and delay.

I never saw anybody do anything useful with packet satellite.

Was it just an experiment?

Yeah. It was an experiment to see if it would be a good idea, and as far as anybody could tell, it wasn't a good idea. So it just kind of quietly went away.

It was Bob Kahn's baby.

I see.

So you know, Larry got a lot of the gop from, as what you described as the lovers of the electron versus -- you know, and he was a lover of the bit, and I put that in the book somewhere. I don't know if you've seen it yet. But once packet switching became more mainstream, these people -- Larry kind of paved the way for like Kahn and Vint Cerf, so let me tell you more.

Size and turf wars gave rise to it?

Dissatisfaction with BBN running the known universe, and realization that not only was BBN doing a bad job, but everybody would do a bad job. My experience is that, and here I was an observer, I was not a participant, but I was a very astute observer paying a lot of attention, and so I believe there is credibility in what I am reporting here, but I'm not a principle, is that there was increasing grumbling about the role of BBN and network operations. Their heavy handedness, their secrecy.

Oh, they wouldn't release this --

Yeah, exactly. And so we have to find a way to change this network so that BBN doesn't have to run it, and people started looking, started a planning process for how do you make a plan for a network that isn't run by BBN? And they very quickly, like matter of hours, came to the realization that nobody else could do any better and that any network of any kind that is centrally controlled by a small organization is going to have problems. So purely from an administrative control, who owns what turf issue, the principle that the network could not be centrally operated came out of dissatisfaction with the central operation by BBN.

very first local network that I ever saw was in about 1977. I think they existed as early as '75, but the major research universities did not have local networking until the late 70's.

Did Ethernet start it all?

Ethernet started it all. Ethernet is what really did it because it gave people something that they needed to connect. If you didn't have a network, then the urge to connect it to something isn't there. And if your whole university was not connected to Arpanet, then CS Net gave you a way to connect a computer at your university to the Arpanet. So that was all great, but the thing that really provided the forcing function was that Ethernet created tens of thousands of things that people wanted to connect to the Arpanet and weren't willing to make part of it. And all of a sudden one day the Arpanet was the tail wagging the dog. Arpanet by 1980 had grown very accustomed to its position of center of the known universe. *"We're Arpanet and you're not. We are centrally controlled. We have subnets. We have this. We have that. You need a letter from God to join."* But by 1980, the number of computers in the world that were connected to Ethernets here and there was vastly greater than the number connected to Arpanet.

Why did these guys want to get onto the Arpanet?

Because it was a status thing. Real universities were on the Arpanet and up and coming universities weren't.

Also one thing that people have said to me is if you were in computer science and you weren't on the Arpanet --

You were nobody.

Right.

Yeah, it's true. Furthermore, at Stanford, for example, which is where I was in 1980, there were exactly four computers on campus that were allowed to be on the Arpanet, because that's how many connections Stanford had. There were more than four groups at Stanford that believed that they were worthy. What is this? [walks

away -- tape is turned off and back on] -- attribute of the Arpanet was that you had a contract from Arpa to connect to it. You couldn't plug in unless you had a license to connect from the Defense whoever it was.

So you had to be doing some kind of Arpa work.

Yeah. But most universities started to develop communities in which the Arpa contractors were in a minority position. Universities didn't like having privileged classes. So for example, at Stanford, the Electrical Engineering Department was ten times the size of the Computer Science Department. It had more money, more prestige, more deans. I mean, the EE Department was big time and the Computer Science Department was not. But the Computer Science Department had the Arpanet connection and wouldn't let EE near it. So the Stanford EE Department, from which John Shock came, was very strongly, powerfully motivated to topple the control structure of the Arpanet in order that they could get a piece of a larger pie. So the exclusivity of the Arpanet Club combined with the sudden availability of local area networking created, all of a sudden out of nowhere, huge numbers of people who weren't allowed to connect to the Arpanet and who didn't like that.

Is that what gave rise to CS Net?

Yes. CS Net was designed to provide a way for whole universities that were unable to connect to the Arpanet and companies to connect, and its goal was to get every computer science department in the country connected, even at universities that didn't have Arpa contracts. But all of a sudden with the construction of CS Net, people realized that you didn't really need to be on the Arpanet so much as you needed to be able to talk to people who were on the Arpanet. That was the way that the pin broke the bubble was it made people realize that they were asking for the wrong thing. That being on the Arpanet was not the big deal, but being able to act as if you were on the Arpanet was the big deal. So CS Net allowed everybody to understand that a bunch of little networks would do, as long as they were all connected to each other. You didn't need to have one big network in the sky. That several separate networks would solve the problem. Ignoring the whole status thing which was largely crumbling anyhow because of things like the Stanford EE situation.

It all happened so fast that it's really hard for me to reconstruct causality. But philosophically I think it was the backlash of the previously privileged class against the sudden new privileged class of Arpanet people to find someway to force this club to take new members. The fact that it happened at the same time that the 255 was running out, happened at the same time that Ethernet was coming on board, happened at the same time that CS Net was happening, etc., was just serendipity.

Was each dependent on the other?

All of them provided additional pressure for the collapse of the Arpanet management structure. The minute that you let anything that is not Arpa connect to the Arpanet you might as well give up, because you no longer have the exclusive club, and that was what CS Net did.

Then after CS Net, came NSF Net.

Well, no. NSF Net was what they did to Arpanet to get it the hell out of Arpa and recover some dignity. NSF was simply a holding tank to put Arpanet into as it gradually disintegrated.

Right.

The main reason for that was Washington based. Having to do with the allocation of money from Arpa versus allocation's money from NSF. The Arpanet was all built with 6.1 money, if you know what that means. Under Title 6, there's 6.1, 6.2, and 6.3. 6.1 money is basic research. 6.2 money is applied research. 6.3 money is the construction of weapons. Each one of them has about a factor of a $100 more than the one before it. The moving of Arpanet to NSF was basically a recognition that it was time to spend 6.1 money on something else.

So in this whole process, the Arpanet is sort of fading.

Well, the Arpanet is fading, but not in its own mind. There is this wonderful twisting of a quote that I'm trying to remember here. He's a something in his own mind. I forget. It will suddenly come to me. Most of the Arpanet community, especially BBN, was so inwardly focused that they didn't notice any of this

happening. That the growth of what was going to become the Internet around them was this tremendous hurricane happening outside that they didn't even see, because they were still seeing the world as being centered around the Arpanet.

They probably still do.

Still do? I don't know.

I don't know. What I'd like to do is portray how that happened.

But nothing happened and that was what happened. It's easy to describe an event. It's much more difficult to describe a non-event or a non-happening. What happened was that the Arpanet, largely because of its central control, became a dinosaur which is another centrally controlled animal, and it was not able to evolve as fast as the rest of the market and industry evolved. So the Arpanet grew at exactly the pace that they had planned for it, in exactly the direction that they had planned for it. But the rest of the world grew five times faster and just swept by it. So the very things that put the Arpanet out front in the beginning were the things that doomed it in the end.

You mean the very central --

The very central control and the subnet of IMPs. Because if you didn't have an IMP, and where were you going to get one, you couldn't connect.

So people found alternatives.

People found alternatives. They'd look at the Arpanet and said, *"Oh, that's a nice club. I wish I could join. But I can't, so I'm going to make my own club."* Then CS Net found a way to connect their own club to the Arpanet, so that it didn't really matter whether they were on the Arpanet or not, for anything except prestige. Then I'm missing a critical event here in terms of who did it.

Who did what?

Let me tell you what the event is. In 1987 or thereabouts, it became possible for

corporations that didn't have an Arpa contract to connect to the Internet.

1987?

Yeah. Corporations that didn't have an Arpa contract. Or maybe it was a little later than that. There's this government document that I've actually signed. I think it's called and LTC, a License To Connect, and it's a piece of federal government deeds that you need to show the officer controlling an IMP before you can actually make a connection to an IMP. I have signed one for the IMP at Nasa Ames for Digital, on behalf of Digital. But then somewhere in there, somebody realized that connection is transitive and if you are connected to the Arpanet and I am connected to you, then I am connected to the Arpanet. It suddenly no longer mattered.

You know who the person who had the best perspective on this would be? Is Marty Schoffstall. Do you know him?

No.

He is the guy who forced Arpanet's hand at letting commercial companies connect.

Oh, really. Where is he?

He's an obnoxious SOB. He owns PSI. His name is S-c-h-o-f-f-s-t-a-l-l. But it is Marty's persistent bullying of Arpa that forced them to give up. He was involved in all of this and he has strong opinions and good memories. So if you call him --

Where is PSI?

They're in McClain, Virginia. 703 area code. PSI is actually headquartered in Troy, New York, but Marty's office is in McClain, Virginia. He is the President and CEO of PSI, Performance Systems International, and he founded that company basically by bullying Arpa into letting him connect by lobbying through Congressmen and Senators. He is the one who really caused it to happen. I wouldn't go so far as to give him credit for doing it. But him being a schmuck and constantly being all over Arpa's case to let him have what he wanted is what really knocked the whole thing down. So if you're going to write about the birth of the Arpanet, you might want

to write about the death of the Arpanet, and he is very definitely the guy who manufactured the gun that killed that killed the Arpanet.

-- is decommissioned.

That only happened a couple of years ago.

Right. 1990. Mark Cohen was the one that pulled the plug. That's it. That's the end of the book and then the epilogue about the reunion.

You need two epilogues. What was the date of the BBN 25th Anniversary Party? October '94?

Yeah, it was September 10, 1994.

Okay. Call Microsoft Network and ask them what they were doing on September 10, 1994. Call Rick Adams, CEO of UU Net Technologies, and ask him what he was doing on September 10, 1994. Put in, *"While this party was taking place, the following big Internet businessmen were doing such and so."*

Find out what Tim Burner's ... was doing?

Something like that. Out with the old and in with the new. But at the same time, September 10, 1994 that BBN was having this party, it would be really cool to find out what the big guns of the Internet were doing. There are five companies that dominate the Internet, and of those, the most colorful CEO is Rick Adams, the most obnoxious is Marty Schoffstall, and none of them was invited to this party at BBN.

What does UU Net do?

UU Net is an Internet service provider. They sell Internet dial tone. They are 10,000 times bigger than the largest that the Arpanet ever got.

Oh, wow. That's great. That's a great idea. You're right. None of them is entitled.

Right.

And who else?

There are five companies that dominate the Internet. They are UU Net Technologies of McClain, Virginia, whose CEO is Rick Adams. They are Performance Systems International, PSI, headquartered in Troy, New York, whose CEO is Marty Schoffstall. There is Advanced Network Systems of Ann Arbor, Michigan, recently bought by America On-Line, and therefore, their CEO is now Steve Case, I guess. There is Network MCI, for which Vint is probably the closest thing to a CEO. But Vint has an actual boss at Network MCI. Their operations staff is somewhere in Rockville, Maryland; something like that. I forget. Then AT&T, which owns BBN Planet -- no, sorry. BBN spun off the BBN Planet Corporation trying to make a buck out of this, but they haven't really cracked the big time.

BBN bought these regionals.

Yeah. But they forgot to buy Transit and so they blew it.

They forgot to buy who?

BBN doesn't own bandwidth. All they own is regionals. So they are dependent on people who own city to city wires.

Who owns those wires?

The companies that I just named for you.

They own the wires?

Yeah. The white board in my office has this list on it. So when we go back to my office, I'll go over it with you. The Internet is officially two or three million nodes and those five companies, among them, are responsible for about 2.2 million nodes connected to the Internet, where AOL counts as one node and Zilker counts as one node and so on.

So how about how the regionals emerged. How did that happen?

Regionals emerged as part of the process of trying to keep some control by giving away a measured amount of control and seeing if that helped. So the NSF Net created these 13 regionals, each one of which had the exclusive franchise in that region to connect to the NSF Net. So for example, the local one here in the San Francisco Bay area was Barnet which originally started out at Stanford. In the beginning, you could not connect to the NSF Net backbone unless you connected through Barnet. What killed it was companies like Alternet saying, *"Who needs it?"* Buying their own cross-country bandwidth, setting up a parallel network that has just as many wires as the NSF backbone, and letting anybody with money connect. So all of a sudden, the thing that Barnet was granted an exclusive franchise to no longer mattered, because other companies were able to deliver the same service without the exclusive franchise and didn't have to pay the licensing fee.

So you just lost me. Here we are in parallel. We've got the NSF backbone. Then we've got these regionals. Then you have something like Alternet bypassing the whole thing?

Yeah. ANS bypassed the whole thing originally and then Alternet figured out that they could do it too, and then PSI did it too. So then the commercial Internet exchange was formed to circumvent. This is another book, because this is unbelievably complicated. Very tricky and very forceful, and I think it's a diversion for your book on your deadline to talk about this now. However, this is a great topic for another book. This is the most purist form of cut-throat capitalism I have ever seen in my 45 years. This is Wild West stuff. It's happening right now. I'm in the middle of it. I'm having a ball. But this time I'm a principle, instead of just an observer and a participant.

So tell me about the regionals. The emergence of the regionals is important.

The emergence of the regionals was sort of like a lumpectomy. It's can we keep the patient alive by cutting off one breast or something. It appears that NSF Net is dying. It appears that Arpanet is dying. Can we keep the concept alive by creating these 13 regionals? Is that enough of a concession that people will stop pushing for concessions? So when the thrust against central control got to be too powerful, NSF, Steve Wolf is the guy who did this, responded by creating the 13 regionals. I guess, originally there were 11 and then two more were added later. To see if that would decentralize it enough that people would stop wining and complaining.

Stop wining about what?

Central control. About exclusive club. About nobody can connect unless such and so.

The story I've heard from these guys is that NSF said, *"We can't afford to build a nationwide network. You guys are going to have to kick in."*

That's part of it. But the real reason that it started at all, if they were even willing to consider it, was to shut people up. And Marty Schoffstall was the chief person that they were trying to shut up. You really need to hear his perspective on this. I would be surprised if Marty Schoffstall had any idea who I am. It's not in his nature to remember people. By and large, I suspect that I am relatively invisible to most of the people you talk to. I would be really surprised if very many of the people that you have interviewed for this book had ever even heard of me.

That's not true at all.

You've seen the movie Star Wars?

No.

In the genre of serial fiction, the standard way to end something is with a teaser opening paragraph of the next whatever. So the closing scene of the movie Star Wars consists of the bad guy getting into a rocket ship and racing away undetected. So what this says is there's more to come. The bad guy is still alive and you'll have to see the next movie to find out what happens to him. I believe the technique was first invented for the serial radio programs of the 1930's and 40's where at the end of each *The Shadow Knows* there would be two sentences about what happened at the beginning of the next one to try to get people to tune in next week. Buck Rogers Radio Show did the same thing.

So at the end of the book where the last node on the Arpanet is unplugged, to have one paragraph about the 700,000 customers of UU Net Technologies, Inc. or something like that, you know? Or interview the guy who was on the night shift help desk of UU Net that night to find out how many trouble calls he logged, or how